DoOR:

Database of Odorant Receptors
User's Guide

Preliminary Version

Authors

Shouwen Ma, Daniel Münch, Martin Strauch, Anja Nissler, C. Giovanni Galizia

Zoology and Neurobiology
University of Constance, Germany

August, 2009

A free open-source software

preliminary version

Contents

Document conventions …….....3

1 Introduction …….....4

2. Preliminaries …….....5
2.1. Installation …….....5
2.2. Get Help …….....5

3. Quick Start …….....6
3.1. Access odorant receptor data ……..……6
3.2. Access supported data …….……8
3.3. Data visualization ……..9
3.4. Odorant responses estimation ……15
3.5. Back Projection ……19

4. Operation …….....22
4.1. Integration approach ……22
4.2. Mapping receptors into database ……28

5. Extension …….....32
5.1. Import the data and update the supported data ...…… 32
5.2. Update response matrix ……35
5.3. Build packages ……36

6. Acknowledgment ...……..…….....40

7. References ……...…….…….....40

preliminary version

Document conventions

Fonts	Example	Index
Times New Roman	Introduction	Text
Italic	*Or22a*	Species, receptor names
Courier New	`?DoOR.function`	R commands
Courier	`$sudo`	Linux

“>” : R command line
“#”: R comment (not executed)
“$”: Linux command line

preliminary version

1. Introduction

DoOR is the Database of Odor Responses that integrates *Drosophila* odor response data from different sources. The DoOR algorithm for integrating odor responses is described in [PUBLICATION].
DoOR is available as two libraries for R. DoOR.data contains the actual database and DoOR.function comprises R functions for visualization and creating the database using the DoOR algorithm.

Up-to-date packages can be obtained from http://neuro.uni-konstanz.de/DoOR

This guide provides an introduction to the main features of DoOR, a quick start navigation to show how to use R functions and the principle of data reconstruction. Finally, the guide will show you how to introduce your own data into DoOR.

For more detail, and an overview of the DoOR Project, see {link to publication} and http://neuro.uni-konstanz.de/DoOR. If you are only interested in the data, you can access odor-response profiles directly at that website, without any need to implement DoOR within R. If, however, you want to add your own datasets, or you want to use this package for other species, or you want to perform other more advanced features, you will need to install the package on your computer. That is also the case if you want to work on a previous version of the package and/or the data.

preliminary version

2. Preliminaries

2.1. Installation

DoOR is constructed under the R environment. You will need to install R, an free statistics package that is available at http://CRAN.R-project.org. The R archive provides both binary versions and source code. Within R, you will need to install and load two libraries: DoOR.data and DoOR.function.

Windows:

For installing the packages type

```
> utils:::menuInstallLocal()
```

and select the "DoOR.function" and "DoOR.data" zip-files.

Alternatively, use the menu "Packages" – "install packages from local zip files".

This also works for Mac OS 10.5.

Linux::

To install packages under Linux execute the following lines in a command line. You might need administrator rights for the installation, then you could for example add the command `sudo` in front of each line.

```
~/$ R CMD INSTALL DoOR.data_1.0.tar.gz
~/$ R CMD INSTALL DoOR.function_1.0.tar.gz
```

2.2. Load the package

Before using DoOR, you need to load the packages first:

```
> library(DoOR.function)
> library(DoOR.data)
```

2.3. Get Help

preliminary version

General help on using R:

```
> help.start()
```

After loading the DoOR libraries with the library command (see 2.2), help on DoOR-specific topics will be available:

```
> ?DoOR.function
> ?DoOR.data
```

preliminary version

3. Quick Start

3.1. Access odorant receptor data

Load all datasets including the precomputed response matrix and data

```
> loadRD()
```

Show information on a specific odorant receptor (for example *Or22a*) in detail by typing:

```
> ?Or22a
```

or:

```
> help(Or22a)
```

The command ? or help() is used to load the documentation of DoOR function and data. We also integrated receptor information. You will see the description for this receptor; the format of the data; the biological information about the receptor, including sequence location, expression, housed sensillum and neuron, co-expression information, targeted glomerulus and further comments, and references. As an example, take *Or47b*:

Or47b

Description

Response profile of Or47b

Usage

1.

-
-

preliminary version

Or47b

Description

Response profile of Or47b

Usage

```
data(Or47b)
```

Format

A data frame with 251 observations on the following 8 variables.

`Class`
: a factor with levels `acid` `alcohol` `aldehyde` `amine` `arom` `ester` `ketone` `N ring` `O ring` `other` `sulfid` `terpene`

`Name`
: a factor with odorant names

`CID`
: a character vector with compound ID

`CAS`
: a factor with odorant CAS numbers

`Hallem.2006.EN`
: a numeric vector; electrophysiological recording in empty neuron from Hallem et al. 2006 (Hallem and Carlson, 2006)

`Hallem.2004.EN`
: a numeric vector; electrophysiological recording in empty neuron from Hallem et al. 2004 (Hallem et al., 2004)

`Hallem.2004.WT`
: a numeric vector; electrophysiological recording in wild type from Hallem et al. 2004 (Hallem et al., 2004)

`Pelz.2005.Or47bnmr`
: a numeric vector; normalized mean values of calcium imaging recording in wild type from Pelz et al. 2005 (Pelz, 2005)

Details

Or47b

`Sequence location`:
: 2R:7,207,215..7,208,817 [-] (Tweedie et al., 2009)

`Expression`:
: in adult (Couto et al., 2005), (Vosshall et al., 2000)

`Sensilla`:

preliminary version

trichoid sensilla (antenna) (Couto et al., 2005)

`Neuron`:

at4 (Couto et al., 2005), atXA (Hallem et al., 2004);
atXA assigned by Hallem's paper (Hallem et al., 2004), indicates a formalized, tentative nomenclature for trichoid neuron.

`Coexpression`:

no recorded

`Glomerulus`:

VA1v (Couto et al., 2005), VA1lm (Hallem et al., 2004), (Berdnik et al., 2006)
VA1lm is innervated by fruitless-expressing neuron (Manoli et al., 2005), (Stockinger et al., 2005).

`Comment`:

1. Or47b responded to virgin female extracts (action potential 100 <= n < 150 impulses/s) and male cuticular extracts (action potential 100 <= n < 150 impulses/s) but not mated female, male genital material, virgin-female genital material, mated female genital material and 11-cis-vaccenyl acetate (action potential n < 50 impulses/s) (van der Goes van Naters and Carlson, 2007).
2. Three glomeruli (DA1, VA1lm and VL2a) are innervated by fruitless-expressing neuron, likely process sex pheromones during male courtship (Manoli et al., 2005), (Stockinger et al., 2005). Blocking synaptic transmission in these ORNs profoundly reduced male courtship (Stockinger et al., 2005).

References

- (Hallem and Carlson, 2006) : Hallem E. A. & Carlson J. R., 2006, *Coding of odors by a receptor repertoire*, Cell,125:143-160
- (Hallem et al., 2004) : Hallem E.A., Ho M. G. and Carlson J.R., 2004, *The Molecular Basis of Odor Coding in the Drosophila Antenna*, Cell, 117:965-979
- (Pelz, 2005) : Pelz, D., 2005, *Functional characterization of Drosophila melanogaster Olfactory Receptor Neurons*, 2005, Dissertationen online, http://www.diss.fu-berlin.de/2005/335/index.html, Freie Universitaet Berlin

Examples

```
data(Or47b)
```

Show the data:

```
> Or22a
```

	Class	Name	CID	CAS	Hallem.2006.EN	Hallem.2004.EN	Hallem.2004.WT	Pelz.2006.EC50	...
1		SFR	<NA>	SFR	4	7	7	NA	...
2	other	water	962	7732-18-5	NA	NA	NA	NA	...
3	amine	ammonium hydroxide	14923	1336-21-6	17	NA	NA	NA	...
4	amine	putrescine	1045	110-60-1	16	NA	NA	NA	...
5	amine	cadaverine	273	462-94-2	17	NA	NA	NA	...
6	amine	ammonia	222	7664-41-7	NA	NA	NA	NA	...
7	amine	ethanolamine	700	141-43-5	NA	NA	NA	NA	...
8	amine	heptylamine	8127	111-68-2	NA	NA	NA	NA	...

preliminary version

NOTE: The first four columns contain odorant class, name, compound ID (CID) and CAS number. The following columns are response data from different studies.

3.2. Access odorant information

The odorant information are acquired by sending a query to PubChem. The result is displayed in a WWW-browser.

```
> showOdor(CID=222,odor.data=odor)

> showOdor(CAS="64-19-7",odor.data=odor)
```

Some function arguments such as “ORs”, and “odor” are data frames, which are needed for many functions and preloaded.

Supported data

The background data list can be found in the documentation of the “DoOR.data” package by typing:

```
> ?DoOR.data
```

The background data consists of:

AL256	A figure matrix of antennal lobe, used for drawing functional antennal lobe; see ORimage()
data.format	The format of response data
glo.dist	A matrix of distance coefficients between glomeruli
odor	Database of odors
OGN	The map matching receptor to sensillum, to receptor neuron and to glomerulus
ORs	Names and expressions of odorant receptors
reference	References of response data
response.matrix	Response matrix of odorant receptors
response.range	Response ranges for each study
unglobalNorm_response.matrix	An unnormalized response matrix
weight.globNorm	Weight matrix for global normalization

preliminary version

For further details of supported data type:

```
> ?odor
```

or

```
> ?reference
...
```

3.3. Data visualization

Show the response profile for a given odorant receptor:
The data frame should contain five columns: “odor class”, “odor name”, “compound ID”, “CAS number” and data column.

```
> Or47bHallem <- Or47b[,c(1:5)]
> head(Or47bHallem)
  Class              Name   CID       CAS Hallem.2006.EN
1  <NA>               SFR  <NA>       SFR             47
2 other             water   962 7732-18-5             NA
3 amine ammonium hydroxide 14923 1336-21-6             39
4 amine         putrescine  1045  110-60-1             22
5 amine         cadaverine   273  462-94-2             31
6 amine            ammonia   222 7664-41-7             NA
```

The response values can be both recording values and consensus values. If the data is not consensus data (in range of [-1, 1]), only two colors, blue and red are used for indicating positive and negative response, respectively. If the data is consensus, a color ramp will be used for coding response intensity. Two examples show PlotChemicals for *Or47b*. The spontaneous firing rate(SFR) were subtracted from response values for both cases.

preliminary version

```
> data(Or47b)

> Or47bHallem <-
Or47b[,c(1:5)]

#substract spontaneous firing
value from odorant response
values

> Or47bHallem[,5] <-
Or47bHallem[,5] -
Or47bHallem[1,5]

> PlotChemicals(
Or47bHallem[order(Or47bHallem
[,5],
decreasing=TRUE),],
tag="Name", x.range=c(-40,
20))
```

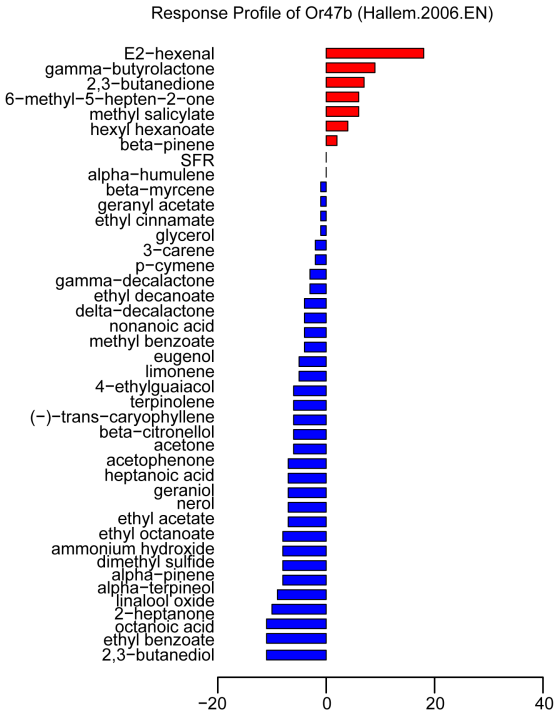


Figure 1.: response profile of *Or47b* measured by Hallem in the empty neuron preparation. Red color codes for positive values and blue for negative values.

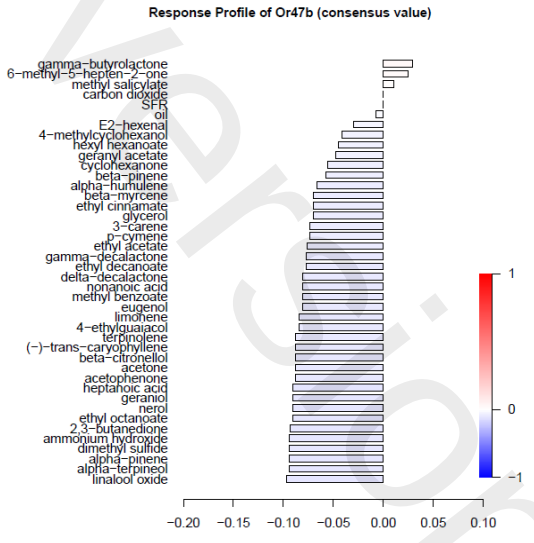
```
> data(Or47b)

> RP.Or47b <- modelRP(Or47b)
> RP.Or47b <-
RP.Or47b$model.response

> sfr <- RP.Or47b[RP.Or47b$CAS
== "SFR",5]

> RP.Or47b[,5] <- RP.Or47b[,5]
- sfr

> PlotChemicals(
RP.Or47b[order(RP.Or47b[,5],
decreasing=TRUE),],tag="Name",
x.range=c(-0.3, 0.1))
```



preliminary Version

Figure 1.: Consensus response profile of *Or47b*.

Tuning Breadth

Tuning Breadth is used for visualizing the response spectrum of a receptor. Generalists have a distribution with broad tails, while specialist receptors have a distinct peak that decreases sharply towards the tails.

Show the response spectrum of a receptor:

```
> data(response.matrix) #
load response data

> x<-
response.matrix[,"Or22a"]

> x<-na.omit(x) # omit the
NA entries

> tuningBreadth(x,las=2,
main="tuning breadth of
Or22a",col="lightblue",ylim
= c(0,1),border = NA)
```

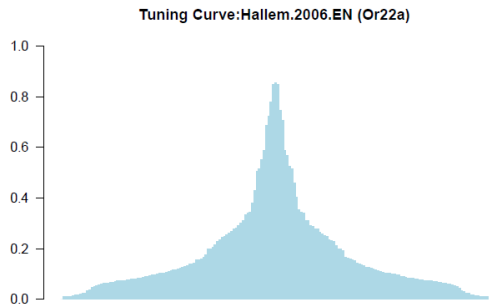


Figure 2.: Tuning curve shows the response profile of *Or22a*.

Compare two studies for an odorant receptor:

The responses of receptor *Or13a* have been recorded in two studies. Data "Kreher.2008.EN" contains data that were measured in an empty neuron preparation by Kreher's study published in 2007. "Schmuker.2007.TR" contains data that were recorded in wild type neurons ab6A by de Bruyne and published as Schmuker.2007.

preliminary version

```
> data(Or13a)

> comparDiagram(x=Or13a,
y=Or13a,
by.x="Kreher.2008.EN",
by.y="Schmuker.2007.TR")
```

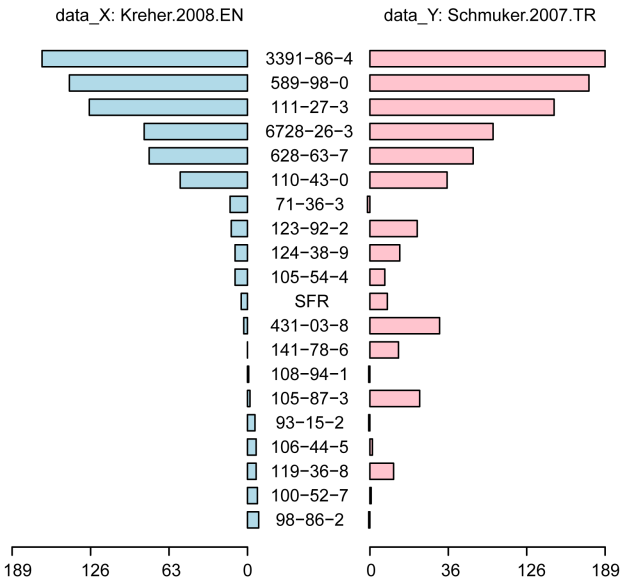


Figure 3.: Comparison of the two studies. The values on the left side were ordered decreasingly, the values on the right side were ordered by matching the CAS number to the left.

Visualize the response profile of odor receptors to given odors.

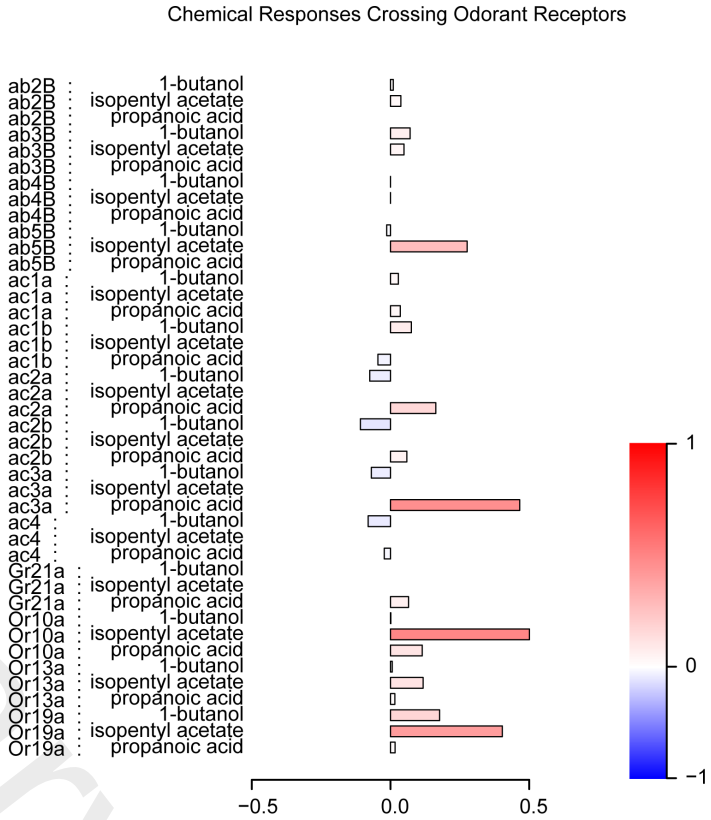
```
> data(odor)
> data(response.matrix)
> cas<-c("71-36-3","123-92-2","79-09-4")
> consResp<-findRespNorm(cas, responseMatrix =
response.matrix)

> head(consResp)    # show first six rows

     ORs     Odor     Response
1   ab2B  71-36-3  0.009879338
2   ab2B 123-92-2  0.037249586
3   ab2B  79-09-4           NA
4   ab3B  71-36-3  0.070703791
5   ab3B 123-92-2  0.048678680
6   ab3B  79-09-4           NA
```

preliminary version

```
> PlotReceptors(consResp
,
odor.data=odor,tag="Name"
)
```



preliminary version

Figure 4.: The color bar from red to blue indicates that the response intensity from most excitatory to most inhibitory. White area between red and blue indicates that the odor response is weak.

Overview of response matrix with dotplot:

The consensus values in the response matrix are normalized within the range from 0 to 1. Because the response profiles of the odorant receptors have been globally normalized, maximum response values of each receptor do not necessarily reach 1, indicating that the best odors may not have been found in some odorant receptors. Inhibition is defined as a value that is lower than the spontaneous firing rate. In order to visualize inhibitory responses, we subtract the spontaneous firing rate from the consensus response values and then renormalize the response spectrum.

$$f(x,SFR)_{resetSFR} = (x - SFR) \cdot \frac{\max(x)}{\max(x) - SFR}$$

where "x" is the vector containing the response values of given receptor and "SFR" is a numeric value indicating the spontaneous firing rate of this receptor.

```
> data(response.matrix)

> resetRM<-apply(response.matrix,2,function(x)
resetSFR(x,x[1])) # apply resetSFR() on each column, the
first value of each column is the value of spontaneous
firing rate.

>
ORdotplot(resetRM[c(150:240),],cex.labels=0.7,type='BW',dot
.size=1.5)
```

preliminary version

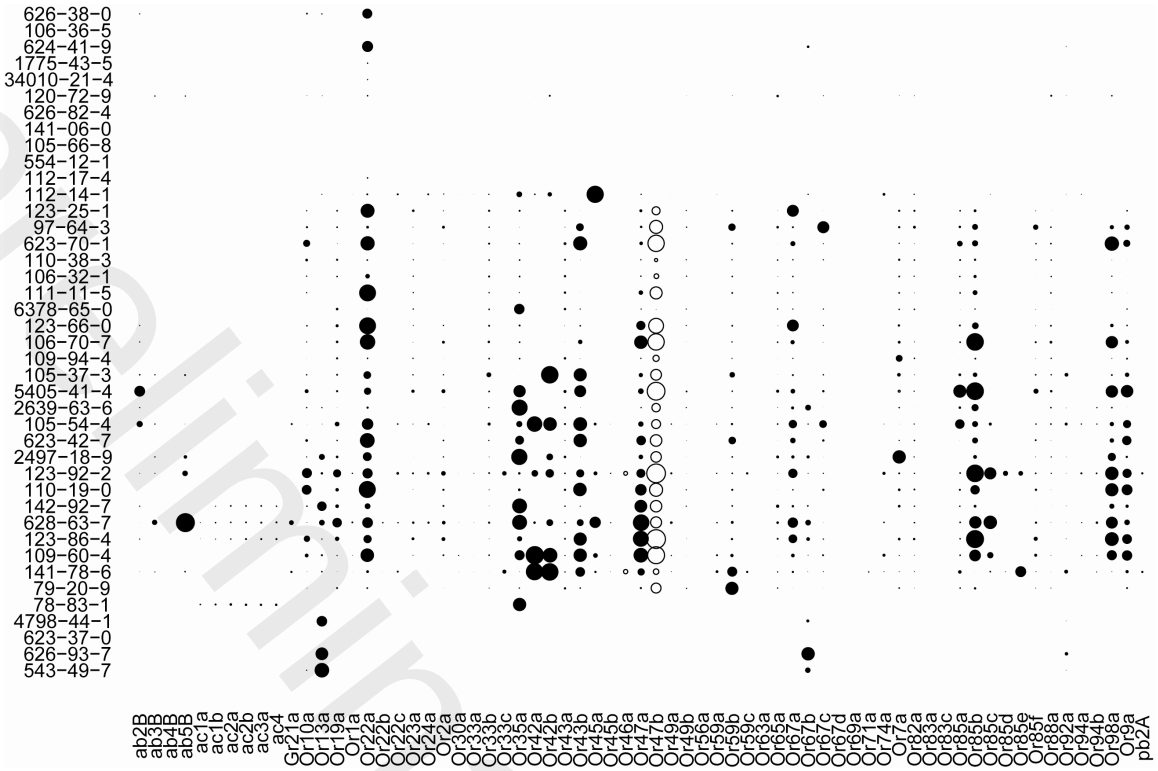


Figure 5.: show the odorant responses across receptors with a dot plot. A blank represents no available data; the size of dot represents the intensity of odorant response; filled circle and open circle response positive and negative values, respectively.

Show the functional antennal lobe response to a single odor

```
> cas<-c("110-43-0")      # 2-heptanone

> data(OGN)     # load OGN data that maps receptors to
glumeruli

> data(AL256)  # load antennal lobe piture.

> data(response.matrix)

> h2ep<-findRespNorm(cas, responseMatrix = response.matrix)
ALimage(h2ep, main = "2-heptanone (CAS: 110-43-0)",
OGN=OGN, AL256 = AL256)
```

preliminary version

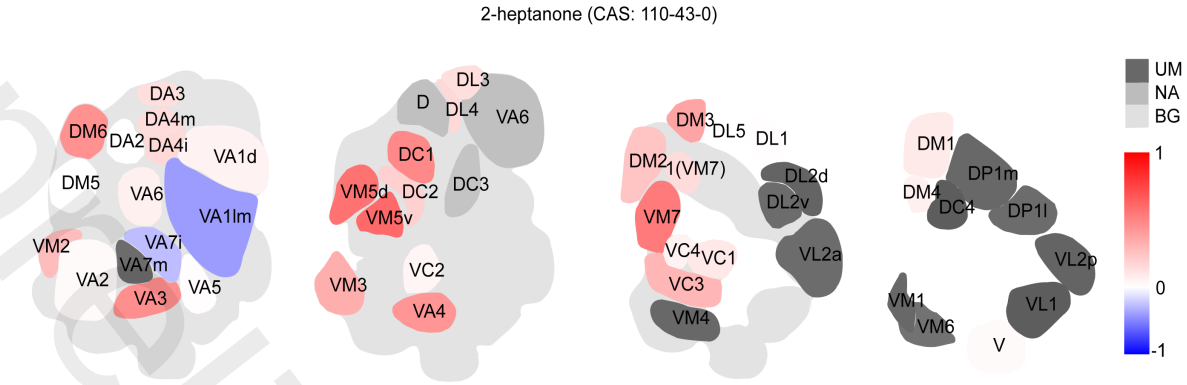


Figure 6.: The functional antennal lobe response to a single odor. Colors go from blue (most inhibitory response) over white (around baseline) to red (most excitatory response). Light grey glomeruli shine through from deeper slices (background, BG). Unmapped (UM) glomeruli are shown in dark grey - in these cases the receptor-glomerulus mapping is currently unknown.

3.4. Odorant responses estimation

Although there are a lot of studies on odorant responses the *Drosophila* olfactome is still quite incomplete. This is due to the complexity of odorant receptors and the close-to-infinite number of odors according to previous studies (Meister, et. al., 2001; Sachse, et. al., 1999), odorants that have similar structure elicit similar response patterns. By following this concept, the odorant response of the target odor can be estimated as a linear combination (Kim, et. al., 2005), in cases where we could find similar odors. There are several approaches to select

similar odors, sorting by chemical and physical properties, coherent odors that have large absolute values of Pearson correlation coefficients and a third party odorant matrix. In most cases, odorant response pattern can not be classified simply by chemical and physical properties, but more satisfactory mapped onto chemical space(Schmuker and Schneider, 2007).

In this study,

we are mining the information of odorant similarity from own response data matrix. In other words to say, we select the similar odors by targeting the numeric feature space of odorant responses that have large absolute values of Pearson correlation coefficients.

Estimating the response of receptor Or22a to ethanol (CAS: 64-17-5)

preliminary version

```
# target receptor and odor
receptor<-"Or22a"
CAS<-"64-17-5"

data(response.matrix)
responseMatrix<-response.matrix

# assign response value NA to target odor, receptor
responseMatrix[CAS,receptor]<-NA

# show the measured value


# a sub-function that is used for finding k odors with the highest
Pearson Correlation coefficient.

      nearest <- function (target, candicate, k)
      {
            N <- nrow(candicate)
            if (missing(k)) { k <- N }
            if (N < k)
            {
                  message("The number of available odors is smaller than
the default (3), so that only available odors will be selected.")
                  k <- N
            }
            # absolute values of Pearson correlation coefficients
between target and candicates
            absCorr     <- abs(apply(candicate,1,function(x)
cor.test(target,x)$estimate))
            sorted_absCorr    <- sort(absCorr, decreasing = TRUE)

            return(sorted_absCorr[1:k])
      }

      responseMatrix          <- as.data.frame(responseMatrix)

      # localize the target receptor and odor in sorted response matrix

      whereTargetReceptor     <-
match(receptor,colnames(responseMatrix))
      whereTargetodor   <- match(CAS,rownames(responseMatrix))

      # non-NA vectors (b ("candicateOdors") and w
("candicateReceptors") ) as candicates

      candicateReceptors      <- which(!
is.na(responseMatrix[whereTargetodor,]))
      Name_candicateReceptors <- colnames(responseMatrix)
[candicateReceptors]
      candicateOdors          <- which(!
is.na(responseMatrix[,whereTargetReceptor]))
      Name_candicateOdors     <- rownames(responseMatrix)
[candicateOdors]

      candi_A <- na.omit( responseMatrix[candicateOdors,
candicateReceptors])
```

preliminary version

```
      # match the odorant location of data "candi_A" to "responseMatrix"
      matchOdor <- match( rownames(candi_A),rownames(responseMatrix) )
      matchReceptor <- match( colnames(candi_A),
colnames(responseMatrix) )

      # vector "w" represents the response values of target odor across
available receptors
      w <- c( as.matrix( responseMatrix[CAS, matchReceptor] ) )
      selectReceptor <- names(responseMatrix[CAS, matchReceptor])
      names(w) <- selectReceptor
      w<-sort(w)
      selectReceptor<-names(w)

      # vector "b" represents the response values of target receptor
across available odors
      b <- responseMatrix[rownames(candi_A), receptor]

      names(b) <- rownames(candi_A)
      b<-sort(b)
      subsetodor<-names(b)

> b
   95-48-7   100-52-7       SFR    64-19-7   138-86-3   124-38-9   111-87-5
0.01134385 0.03116479 0.03391304 0.05070711 0.05625201 0.06253786 0.06800288
  513-85-9  6728-26-3   106-24-1    98-86-2   119-36-8    79-09-4   105-87-3
0.07515067 0.08536189 0.08736033 0.08911544 0.09521574 0.10370175 0.10486154
   67-56-1    67-64-1    97-53-0   141-78-6   111-27-3   431-03-8    71-36-3
0.11791586 0.13079865 0.14174257 0.16653614 0.21616181 0.22948048 0.25896885
  110-43-0  3391-86-4   123-92-2   628-63-7   105-54-4   109-60-4
0.26849417 0.30067607 0.52526482 0.55014927 0.58844967 0.68763607

> w
     Or49b      Or67a      Or23a      Or88a       Or2a      Or98a      Or85f
0.01337175 0.02607130 0.03288981 0.03299875 0.03727293 0.04071159 0.04269889
     Or47a      Or59b      Or43a      Or82a      Or67c      Or65a      Or10a
0.04651597 0.04852848 0.05742629 0.06121915 0.06323438 0.06464458 0.07466297
     Or43b      Or19a      Or85a      Or47b      Or85b       Or9a      Or33b
0.08194255 0.09270386 0.09771639 0.09985090 0.11044907 0.14725503 0.15258732
     Gr21a      Or35a       Or7a
0.20561416 0.22096924 0.22664469


      # sort the data "candi_A" according to vector "w" and "b"
      candi_A <- na.omit( responseMatrix[subsetodor, selectReceptor])

      # the two most similar odors were selected
      nearestodor <- nearest(target = w, candicate = candi_A, k = 2)
>  nearestodor
  67-56-1   71-36-3
0.9584361 0.7480300

selectedOdor <- names(nearestodor)

A <- candi_A[selectedOdor,]
A
            Or49b      Or67a      Or23a      Or88a       Or2a      Or98a
67-56-1 0.02253503 0.02420906 0.01315592 0.04173466 0.05212804 0.04419391
71-36-3 0.03642539 0.17241852 0.19052745 0.05920648 0.12709516 0.08030602
            Or85f      Or47a      Or59b      Or43a      Or82a      Or67c
67-56-1 0.05433386 0.06942072 0.06127587 0.08184591 0.02205249 0.05499538
71-36-3 0.10937467 0.14079881 0.12511124 0.08638887 0.11748797 0.10443982
```

preliminary version

```
            Or65a      Or10a      Or43b      Or19a      Or85a      Or47b
67-56-1 0.06790893 0.05113509 0.08518086 0.08343347 0.08906153 0.10439306
71-36-3 0.05121011 0.03203408 0.31520691 0.26421047 0.05089908 0.08538568
            Or85b      Or9a      Or33b     Gr21a     Or35a      Or7a
67-56-1 0.0688171 0.1605819 0.14964253 0.2056142 0.2381653 0.2029986
71-36-3 0.1766872 0.3957389 0.05647619 0.2754819 0.7847021 0.6133272

b <- responseMatrix[selectedOdor, receptor]

b
[1] 0.1179159 0.2589689
```

Up to here, the linear combination $\begin{bmatrix} \alpha & w \\ b & A \end{bmatrix}$ of two linear systems, $[\alpha \quad w]$ and $[b \quad A]$ has been formed. They share the same coefficients in two the linear systems. In order to estimate the "α", firstly, we solve the coefficients "x" of following linear equation:

$$A \cdot x = b$$

where,

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

X is the coefficients of the linear system, which is also applied in the linear system $[\alpha \quad w]$.

As a result, "α" as a responser can be estimated by multiplying the predictor "w" with coefficients "x":

$$\alpha = w \cdot x$$

In fact, we could not always find the precise coefficients from a complex linear system. We could however find a coefficient vector, that brings A * x as close as b

$$||A \cdot x - b||^2$$

The pseudoinverse of a matrix is used to solve a vector x:

$$x = A^{+} b$$

The "α" can be estimated:

$$\alpha = w \cdot A^{+} \cdot b$$

```
alpha <- t(w) %*% PseudoInverse(A) %*% as.matrix(b)

> alpha
           [,1]
[1,] 0.1173506

# estimate the response value using DoOR function LLSIestPC()
LLSIestPC(CAS = CAS, receptor = receptor,responseMatrix
```

preliminary version

```
=responseMatrix,nodor = 2 )

$estimation
          [,1]
[1,] 0.1173506

$selected.receptors
 [1] "Gr21a" "Or10a" "Or19a" "Or23a" "Or2a"  "Or33b" "Or35a" "Or43a"
"Or43b"
[10] "Or47a" "Or47b" "Or49b" "Or59b" "Or65a" "Or67a" "Or67c" "Or7a"
"Or82a"
[19] "Or85a" "Or85b" "Or85f" "Or88a" "Or98a" "Or9a"

$selected.odors
  67-56-1   71-36-3
0.9584361 0.7480300

# show the measured consensus value
response.matrix[CAS,receptor]
[1] 0.1179159
```

Here, the result shows as a list giving the estimated value, the selected receptors and the selected odors with absolute values of Pearson correlation coefficients to the target odor "*Or22a*".

There is a wrapper function available for estimating all NA entries in a response data.

```
# example

data(response.matrix)
est_data <- DoOREst(da=response.matrix, nodor = 2)
```

preliminary version

$$NRMSE = \frac{\sqrt{\frac{\sum (guess_i - analysis_i)^2}{n}}}{\sigma_{analysis}}$$

3.5. Back Projection

The consensus response matrix containing the normalized data [0, 1] allows us for theoretical analysis of olfactory coding. From an experimentalist's point of view, the odorant response values are more useful if they are given in spikes/sec or fluorescence change. Therefore, we back project the merged dataset onto the original datasets.

We take *Or22a* as example. Currently, this is the receptor with the most responses available and it has also been measured with many different techniques, such as single sensillum recordings and calcium imaging:

preliminary version

```
> data(response.matrix)
> data(Or22a)
> cons_Or22a <- response.matrix[,"Or22a"]
```

Combine the vector cons_Or22a to the "data.format"

```
> data(data.format)

> RP.Or22a<- data.frame(data.format, merged.data =
cons_Or22a[match(data.format$CAS, rownames(response.matrix)
) ] )     # combine two data accordingly to the odors
```

Project the data RP.Or22a back to the normalized mean response data which was measured with calcium imaging:

```
> bp.data <- Or22a[c(1:4,13)]      # first columns of
bp.data contain odorant information, the 13th column is the
recording values from "Pelz.2006.AntEC50".

> dec50 <- backProject(cons.data= RP.Or22a, bp.data=
bp.data, tag.odor= "CAS",tag.cons.data= "merged.data",
 tag.bp.data="Pelz.2006.AntEC50")


# join two coordinate vectors that represent the projection
of consensus response of "propyl acetate" onto fitted
curve.

> x01 <- c(DoORnorm(RP.Or22a[,5])
[which(RP.Or22a$Name=="propyl acetate")],
DoORnorm(RP.Or22a[,5])[which(RP.Or22a$Name=="propyl
acetate")] )


> y01 <- c(-0.2,
     dec50$output[which(dec50$output$Name=="propyl
acetate"),"projected.Y"] )

> lines(x=x01, y=y01)

# Draw a line with arrow that points the back-projected
value

> arrows(x0 = x01[2], y0 = y01[2], x1 = 1.02, y1 = y01[2])

# draw the tricks and labels that represent the
```

preliminary version

```
backprojected scale.

labels2 <- seq(-0.2, 1, length = 7)

# The scale of backprojection is linear correlated to the
scale of normalized measured data.

labels4 <- dec50$rescale[1] + labels2*dec50$rescale[2]
axis(4, at = labels2, labels = labels4)
```

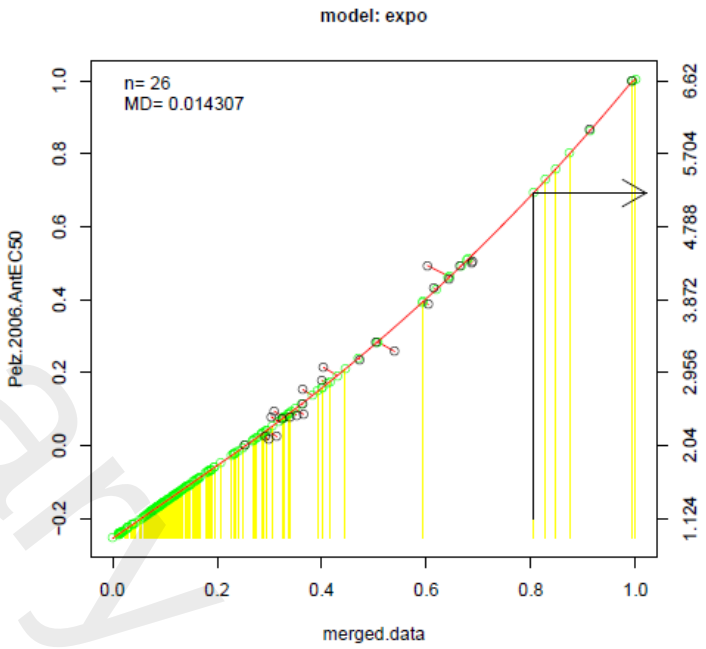


Figure 10.: Back projection. The consensus data was put on x; data "Pelz.2006.AntEC50" .was put on y. Yellow lines indicate the odorants that have not been measured in data "Pelz.2006.AntEC50", and will back projected data. The backprojected scale is shown on the right.

```
> dec50
$rescale
Intercept     Slope
     2.04      4.58

$output
Class     Name     CID      CAS  Pelz.2006.AntEC50 consensus.value  projected.Y   bp.data
amine  ammonia     222 7664-41-7                NA     0.081211965 -0.136004576 1.4170990
```

preliminary version

The result is a list containing $rescale and $output. The rescale parameters were estimated by computing the regression between unnormalized and normalized data of back-projected data ("Pelz.2006.AntEC50"). Note that, the data Pelz.2006.AntEC50 have been normalized to the range [0, 1] before plotting against the consensus data, so that the back-projected data will be rescaled by using following function:

$$bp.data = Intercept + Slope \cdot ¿$$
¿

In the output section, the first four columns contain the odorant information. Ammonia was not measured in Pelz.2006.AntEC50, indicated by NA. The consensus responses is 0.081211965, the back projected normalized mean response is 1.4170990.

preliminary version

4. Operation

4.1. Integration approach

Heterogeneous dataset integration is a process by which two datasets of odor responses for the same receptor are brought together to a consensus data that pools the two datasets into one and thus collects the information of both .

The precondition of heterogeneous dataset integration is that two datasets have a sufficient number of common odor responses. For example, the two datasets Pelz.2006.nmr and Hallem.2006.EN were acquired by measuring the same odorant receptor *Or22a* in *Drosophila melanogaster*. Because several odors were shared by the two studies, their merger is possible.

STEP 1:

Fitting two datasets onto each other using least squares regression

The available linear and nonlinear regression is performed by minimizing the sum of squared distances between the line and the observations. The distance is not measured orthogonally but vertically, which means the regression is not symmetric. We try to fit the data with five models (linear exponential and three non linear models) and their inverse models. The parameters will be estimated by the generic R fitting functions `lm()` and `nls()` for the linear model and the nonlinear models, respectively. The parameters of the inverse functions are estimated by interchanging the variables.

Linear

$$f_{linear}(x) = a + b \cdot x \tag{1}$$

Inverse linear

$$f_{inv.linear}(x) = -\frac{a}{b} + \frac{1}{b} \cdot x \tag{2}$$

Exponential

$$f_{\exp}(x) = a + b \cdot e^{c \cdot x} \tag{3}$$

Inverse exponential

preliminary version

$$f_{inv.\exp}(x)=\frac{\log(\frac{x-a}{b})}{c} \tag{4}$$

Sigmoid

```
# Asym / 1 + exp( (xmid -x) / scal )
```

$$f_{sigmoid}(x)=\frac{Asym}{1+e^{\frac{\text{xmid - x}}{\text{scal}}}} \tag{5}$$

Inverse Sigmoid

```
# xmid - scal * log((Asym / x) - 1)
```

$$f_{inv.sigmoid}(x)=\text{xmid - scal}\cdot\log(\frac{\text{Asym}}{\text{x - 1}}) \tag{6}$$

asympOff

```
# Asym * (1 - exp( - exp(lrc) * (x - c0)))
```

$$f_{asympOff}(x)=\text{Asym}\cdot(1-e^{-e^{\text{lrc}}\cdot(\text{x - c0})}) \tag{7}$$

Inverse asympOff

```
# c0 - log(1 - x/Asym)/ exp(lrc)
```

$$f_{inv.asympOff}(x)=\text{c0 - }\frac{\log(\frac{1-\text{x}}{\text{Asym}})}{e^{lrc}} \tag{8}$$

asymp

```
# Asym + (R0 - Asym) * exp( - exp(lrc) * x)
```

$$f_{asymp}(x)=\text{Asym}+(\text{R0 - Asym})\cdot e^{-e^{lrc}\cdot\text{x}} \tag{9}$$

Inverse asymp

```
# log((x-Asym)/(R0-Asym))/exp(lrc)
```

preliminary version

$$f_{inv.asymp}(x) = -\frac{\log\left(\frac{\text{x - Asym}}{\text{R0 - Asym}}\right)}{e^{lrc}} \tag{10}$$

STEP 2:

Select the best fitting model from ten optional models based on correlation coefficients

Now we have a model function:

$$Y = f(X) \tag{11}$$

STEP 3:

Project the observation points onto the fitted line minimizing the distance between the observation point and the projected point.

$$\min \sqrt{(x_{obs} - X)^2 + (y_{obs} - f(X))^2} \tag{12}$$

then the optimum can be found by following equation:

$$f'\left(\sqrt{(x_{obs} - X)^2 + (y_{obs} - f(X))^2}\right) = 0 \tag{13}$$

We take the derivative with respect to 'X'. The result shows as (calculation details are shown in appendix):

$$\frac{-1 \cdot (x_{obs}) + X + \left[f(X) - y_{obs}\right] \cdot f'(X)}{\left[(x_{obs} - X)^2 + (y_{obs} - f(X))^2\right]^{0.5}} = 0 \tag{14}$$

with this equation we can compute the X coordinate so as Y coordinate on the functional line, which's distance to the observation is minimum.

STEP 4:

Compute the distance between two points on the functional line

$$ds = \sqrt{1 + (Y'^2)} \cdot dX \tag{15}$$

STEP 5:

preliminary version

Transfer the distance values in values from 0 to 1.

In DoOR package, three basic functions are used for heterogeneous datasets integration. Function modelfunction() is used for estimating the parameters; cal.model() for choosing the optimized model(step 1 and step 2). Function projectPoints() executes all five steps to produce a consensus set of response values.

Example for heterogeneous dataset integration

Two datasets for *Or22a* are shown (Pelz.2006.AntEC50 and Hallem.2006.EN). The range in Pelz.2006.AntEC50 ranges from 2.04 to 6.62 (negative logarithmic concentration that is necessary to elicit the half-maximal response), while responses in Hallem.2006.EN range from 2 to 260 (these are response frequencies in spikes/s). Different dimensionalities along the axes influence this result (e.g., deviation along the spike-axis would weigh more, because the value ranges are larger). Therefore, each dataset was linearly scaled to a common range [0, 1] using DoORnorm() before mapping.

```
> data(Or22a)

> range(na.omit(Or22a[,'Pelz.2006.AntEC50']))
[1] 2.04 6.62

> range(na.omit(Or22a[,'Hallem.2006.EN']))
[1]   2 260

> tan22a<-
projectPoints(x=DoORnorm(Or22a[,'Pelz.2006.AntEC50']),y=
DoORnorm (Or22a[,'Hallem.2006.EN']))

> tan22a

$Double.Observations
    ID          x         y           X         Y  distance       NDR
1   86 0.02401747 0.4379845  0.05714813 0.4231506 0.4489362 0.2588112
2  137 0.07860262 0.4728682  0.07940243 0.4724867 0.5030594 0.2900131
3  139 0.00000000 0.2519380 -0.02393086 0.2758688 0.2879997 0.1660315
4  143 0.09388646 0.4108527  0.05873606 0.4266848 0.4528108 0.2610449


$Single.Observation
     ID          x          y           X           Y    distance       NDR
15   79 0.38646288         NA  0.38646288 0.824054838 0.987736604 0.569428970
16   90 0.21397380         NA  0.21397380 0.711384795 0.778003000 0.448517799
17   92 0.07860262         NA  0.07860262 0.470727744 0.501127081 0.288899162
18  198 0.49344978         NA  0.49344978 0.841724338 1.096304885 0.632018454
```

A list of two results is given in data tan22a. One is tan22a$Double.Observations, the other is tan22a$Single.Observation. Both results have a same formation. "ID" indicates the original position of data x and y; "x" and "y" indicate the coordinate of observation; "X" and "Y" indicate the coordinate of projected point on the functional line; "distance"

preliminary version

indicates the distances between `(xmin, f(xmin))` and all points on the functional line; "NDR" indicates the normalized distances across all the distance values.

Data 'tan22a$Double.Observations' means, that those odors were taken by both studies, whereas data tan22a$Single.Observation means, that those odors were not tested by both studies but either of them.

Since two data sets have a same odor arrangement. We can address the odorant names according to their ID.

```
> doubObserv_ID <- tan22a$Double.Observations[,'ID']

> doubObserv_data <-
data.frame(Name=Or22a[doubObserv_ID,2],CAS=
Or22a[doubObserv_ID,4],model.response=
tan22a$Double.Observations[,7],
Hallem.2006.EN=Or22a[doubObserv_ID,5], Pelz.2006.AntEC50 =
Or22a[doubObserv_ID,13])

> ordered_doubObserv_data <-
doubObserv_data[order(doubObserv_data[,3],decreasing=TRUE),
]

> rownames(doubObserv_data) <- seq(1:dim(doubObserv_data)
[1])

> ordered_doubObserv_data
```

	Name	CAS	model.response	Hallem.2006.EN	Pelz.2006.AntEC50
14	ethyl hexanoate	123-66-0	0.9315796	228	6.62
13	methyl hexanoate	106-70-7	0.8462074	260	6.00
11	ethyl butyrate	105-54-4	0.6339355	197	4.35
9	isopentyl acetate	123-92-2	0.6017955	236	4.01
7	pentyl acetate	628-63-7	0.5850395	162	4.13
6	butyl acetate	123-86-4	0.5190946	216	3.34
10	E2-hexenyl acetate	2497-18-9	0.4860582	198	3.22
12	ethyl propionate	105-37-3	0.4677935	192	3.12
8	hexyl acetate	142-92-7	0.4112706	176	2.74
5	1-octen-3-ol	3391-86-4	0.2930847	125	2.42
2	1-butanol	71-36-3	0.2900131	124	2.40
4	3-methyl-butanol	123-51-3	0.2610449	108	2.47
1	2-heptanone	110-43-0	0.2588112	115	2.15
3	1-hexanol	111-27-3	0.1660315	67	2.04

Visualize the data of "model.response" and "Hallem.2006.EN" and "Pelz.2006.AntEC50".

```
> op=par(mfrow=c(1,3))

> op <- par(las=2,cex.lab=0.01,cex.axis=0.7)
```

preliminary version

```
> barplot(rev(ordered_doubObserv_data
[,3]),horiz=T,las=1,col='lightgreen',main='model.response')

> barplot(rev(ordered_doubObserv_data
[,4]),horiz=T,las=1,col='lightblue',main='Hallem.2006.EN')

> barplot(rev(ordered_doubObserv_data
[,5]),horiz=T,las=1,col='yellow',main='Pelz.2006.AntEC50')
```

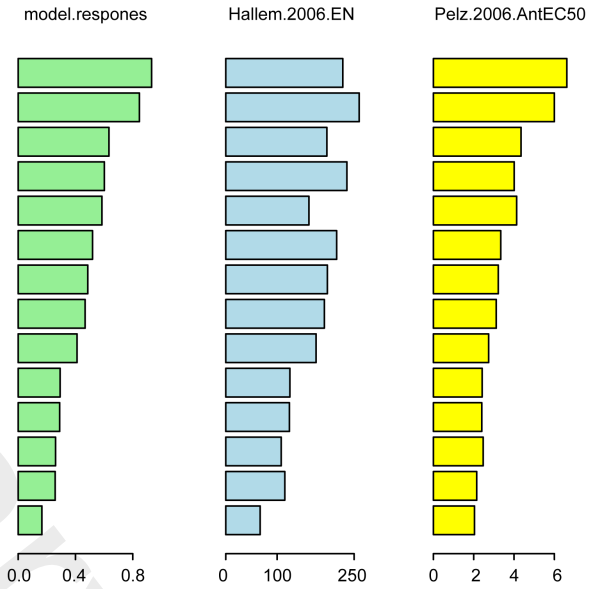


Figure 12.: Comparison between merged model responses and the measured data from Hallem.2006.EN and Pelz.2006.AntEC50.

Nevertheless, not only overlapped odors, but also some odors, which were tested by either of studies, can be generated as consensus data.

Similar process as above, we can address the odorant names according to their ID.

```
> singObserv <- tan22a$Single.Observation[,'ID']

> singObserv_data <-
data.frame(Name=Or22a[singObserv,2],CAS=
Or22a[singObserv,4],model.response=
tan22a$Single.Observation[,7],
Hallem.2006.EN=Or22a[singObserv,5], Pelz.2006.nmr=
Or22a[singObserv,13])

> singObserv_data[10:15,]
```

```
10          ethyl 3-hydroxyhexanoate  2305-25-1    0.297995200           NA         2.43
11                beta-butyrolactone  3068-88-0    0.215621065           NA         2.16
```

preliminary version

```
12              gamma-valerolactone   108-29-2    0.282803313            NA          2.38
13                              SFR         SFR    0.004665498             4            NA
14              ammonium hydroxide   1336-21-6    0.034991236            17            NA
15                       putrescine    110-60-1    0.032658486            16            NA
```

4.2. Mapping receptors into database

Odorant responses of specific receptors can be measured by using transgenic techniques. Kreher and his colleagues expressed Or's in the empty neuron preparation and tested them with a series of odors using electrophysiological recordings (Kreher, et. al., 2008). In addition, Nissler expressed the calcium sensitive fluorescent protein G-CaMP under control of the Or13a promoter in the corresponding neurons. The odorant receptor 13a was first suggested to house in intermediate sensilla (Couto et. al., 2005), whereas Nissler proposed that Or13a houses in basiconic sensilla ab6A.

Neuron ab6A was measured by de Bruyne (de Bruyne, et. al., 2001), using electrophysiological recordings performed on basiconic sensilla without knowing the expressed Ors. Assuming that Or13a is expressed in ab6A, out of the 61 datasets in DoOR, the data from Kreher and Nissler both should match best to the ab6A.

As Or13a is already included into the database, we first have to split the dataset into data coming from single sensilla recordings (“Bruyne.2001.WT” and “Schmuker.2007.TR”) and data that comes from studies recording identified receptor neurons (“Nissler.2007.EC50”, “Nissler.2007.nmr” and “Kreher.2008.EN”).

```
> data(Or13a)

> names(Or13a)

[1] "Class"              "Name"
[3] "CID"                "CAS"
[5] "Schmuker.2007.TR"   "Bruyne.2001.WT"
[7] "Nissler.2007.EC50" "Nissler.2007.nmr"
[9] "Kreher.2008.EN"



> ab6A <- Or13a[,c(1:6)]

> Or13aNMR <- Or13a[,c(1:4,7,8)]
```

then merge those as a consensus data:

preliminary version

```
> RP.ab6A <- modelRP(ab6A,plot=TRUE)$model.response

> RP.Or13aNMR <- modelRP(Or13aNMR, plot=TRUE)
$model.response


> data(response.matrix)

# Assign the response data a new name

> new.response.matrix <- response.matrix

# match odor names between “RP.ab6A” and”
new.response.matrix”

> matchOdor_ab6A <- match(RP.ab6A[,"CAS"],
rownames(new.response.matrix))

# rename the ’Or13a’ to ’ab6A’

> colnames(new.response.matrix)
[which(names(new.response.matrix)=='Or13a')] <- 'ab6A'

# replace response data of Or13a by RP.ab6A

> new.response.matrix[matchOdor_ab6A,'ab6A'] <-
RP.ab6A[,"merged_data"]


> matchOdor_newOr13a <- match(newOr13a[,"CAS"],
rownames(new.response.matrix))
```

The receptors can be sorted into three groups: expressed in adult, larvae and both. We need the data ORs, the second column of “ORs” contains numeric values; 0, 1, 2 and NA indicating expression in adult (0), in larvae (1), both (2) or not recorded (NA), respectively.

```
> data(ORs)

> which_in_adult<-which(!is.na(match(ORs[,2],c(0,2))))

>
selected_ORs=c(as.character(ORs[which_in_adult,1]),"ab6A")
```

We only want to compare the response pattern between Or13a and the receptors that express in the antennal sensilla, so that we are excluding the receptors that are expressed

preliminary version

on the maxillary palp.

```
>
which_in_palp=c('Or42a','Or71a','Or33c','Or85e','Or46a','Or
59c','Or85d',"pb2A")

# since the colname “Or13a” was replaced by “ab6A”

> antenna_ORs=selected_ORs[-c(match(c(which_in_palp,
"Or13a"),selected_ORs))]

> res_Nissler <- numeric()



> for (i in antenna_ORs)
{
     x_Nissler <- RP.Or13aNMR[,"merged_data"]
     y <- new.response.matrix[matchOdor_newOr13a,i]
     xy <- na.omit(cbind(x_Nissler,y))
     if(is.na(which(!
is.na(new.response.matrix[matchOdor_newOr13a,i]))[1]) |
dim(xy)[1]==0)
     # if no data available for selected receptor or no
     overlapped values with the selected receptor, then
     return NA and run next loop
     {
          Rs <- NA
          next
          }

     if (lm(y~x_Nissler)$coef[2]==0|is.na(lm(y~x_Nissler)
$coef[2]))
     # if the two data are fitted horizontally or
     vertically, then return NA and run next loop
     {
          Rs <- NA
          next
          }

     else
     {
          Rs <- cor.test(x= x_Nissler,y=y)$estimate
          }
     Rss <- Rs
     names(Rss) <- i
     res_Nissler <- c(res_Nissler,Rss)
```

preliminary version

```
}

# sort the data in decreasing order
> sorted_res_Nissler <- sort(res_Nissler,decreasing = TRUE)
```

Plot the data after specifying the margin size

```
> par(mar=c(9,4,5,3))

> barplot(sorted_res_Nissler,las=2,ylim=c(-
0.3,1),main="Mapping response profiles of study
'Nissler.2007.nmr' to antennal receptors and
ORNs",cex.main=0.8, ylab = c("Pearson correlation
coefficient"))
```

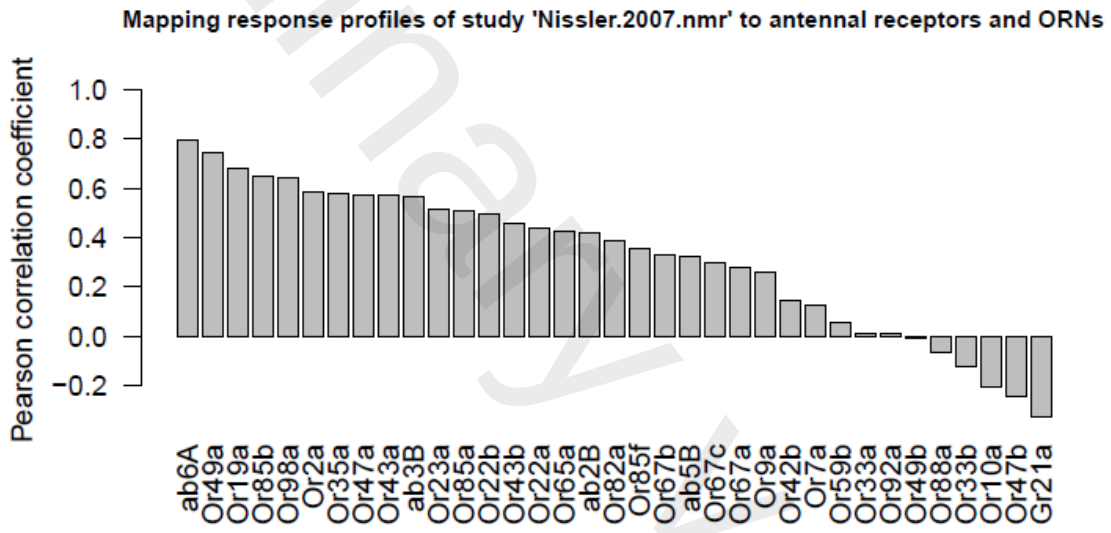


Figure 10.: Mapping response profile of sdataset "Nissler.2007.nmr" to antennal receptors and ORNs.

preliminary version

5. Extension

5.1Importing new data

The general format of odorant response data is shown in following table. The first four columns are set for odorant class (amine, acid etc.), name, chemical identifier (CID) in PubChem and CAS number. The following columns contain the response values in the respective studies.Study names are assigned as follows: for example Pelz.2006.nmr is coded by “Pelz” (the first author), 2006 (published year) and “nmr” (the measurement technique or data character). “nmr” stands for normalized mean responses using the calcium imaging technique. Missing values, i.e. responses not measured in a particular study, are coded with NA

	Class	Name	CID	CAS	Study
1		SFR	<NA>	SFR	
2	other	water	962	7732-18-5	
3	amine	ammonium hydroxide	14923	1336-21-6	
4	amine	putrescine	1045	110-60-1	
5	amine	cadaverine	273	462-94-2	
6	amine	ammonia	222	7664-41-7	
7	amine	ethanolamine	700	141-43-5	
8	amine	heptylamine	8127	111-68-2	
.					
.					
240	other	11-cis Vaccenyl Acetate	<NA>	6186-98-7	

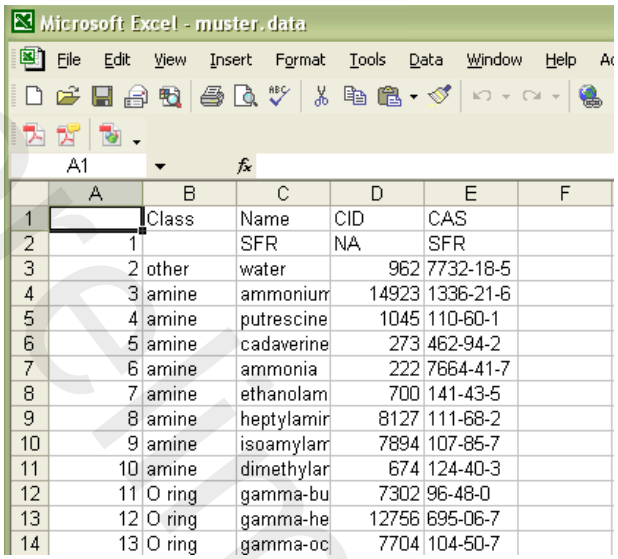
Reading new odorant response data

Assuming that an experiment has been performed with one or more odorants, each odor has a response value or no recording (NA). The first step for integration into DoOR is usually to create an input file in .txt or in .csv format. The input file should contain one oder per row including columns for the chemical name, CAS number and the response value. The CAS number is needed to unambigously identify the odor, because there are multiple names for a chemical.

You can export a data template from the DoOR package:

```
> write.csv(data.format,'data.format.csv')
```

preliminary version



	A	B	C	D	E	F
1		Class	Name	CID	CAS	
2	1		SFR	NA	SFR	
3	2	other	water	962	7732-18-5	
4	3	amine	ammonium	14923	1336-21-6	
5	4	amine	putrescine	1045	110-60-1	
6	5	amine	cadaverine	273	462-94-2	
7	6	amine	ammonia	222	7664-41-7	
8	7	amine	ethanolam	700	141-43-5	
9	8	amine	heptylamir	8127	111-68-2	
10	9	amine	isoamylam	7894	107-85-7	
11	10	amine	dimethylar	674	124-40-3	
12	11	O ring	gamma-bu	7302	96-48-0	
13	12	O ring	gamma-he	12756	695-06-7	
14	13	O ring	gamma-oc	7704	104-50-7	

Figure 13.:

After you have entered the response values, reimport your response data in DoOR.

```
> Orx <- read.csv('data.format.csv')
```

In case you want to combine your data into one of receptor response data:

```
> newdata <- data.frame(CAS=c("111-1-1","222-2-2","71-23-
8","333-3-3"),
        newdata.2009.nmr=c(0.4, 0.5, 0.6, 0.7))

> newdata

      CAS newdata.2009.nmr
1 111-1-1              0.4
2 222-2-2              0.5
3 71-23-8              0.6
4 333-3-3              0.7
```

The new data contains 4 odors. Up to now only odor “71-23-8” is included in DoOR. To combine `newdata` into the Or22a dataset the DoOR function `combData()` is used:

```
> data(Or22a)

> new.Or22a<-
combData(data1=Or22a,data2=newdata,by.data2="newdata.2009.n
mr")
```

Show the data “new.Or22a” only with row 136, and new added rows from 241 to 243 by columns “Name”, “CAS” and “newdata.2009.nmr”.

preliminary version

```
>
new.Or22a[c(136,251:254),c('Name','CAS','newdata.2009.nmr')
]

         Name     CAS newdata.2009.nmr
136 1-propanol 71-23-8              0.6
241       <NA> 111-1-1              0.4
242       <NA> 222-2-2              0.5
243       <NA> 333-3-3              0.7
```

We take the response data that was measured on the antennal lobe (Root, et. al., 2007). The responses indicate the values of fluorescence change.

```
> Root.2007.ER <- data.frame(Name=c("Isoamyl acetate","1-
hexen-3-ol","4-heptanol","3-octanone","Benzyl
acetate"),CAS= c("123-92-2","4798-44-1","589-55-9","106-68-
3","140-11-4"),
     Or10a=c(70,0,0,NA,NA),
     ab5B=c(81,0,0,NA,NA),
     Or22a=c(130,0,12,NA,NA),
     Or43b=c(127,97,90,NA,NA),
     Or13a=c(2,48,0,NA,NA),
     DP1m=c(21,NA,NA,164,12),
     Or42b=c(104,NA,NA,111,32),
     Or59b=c(106,NA,NA,90,108) )

> write.table(Root.2007.ER,"Root.2007.ER.txt")

> loadRD()

> importNewData(file.name="Root.2007.ER",
file.format=".txt", dataFormat  = data.format,
weightGlobNorm  = weight.globNorm, responseRange =
response.range, receptors = ORs)

Data Version 1.0
Date: Nov 04 2009
Function Version 1.0
Date: Nov 09 2009
[1] "DP1m has been added into 'weight.globNorm'."
[1] "589-55-9 is a new odor. Data frames 'odor' and
'data.format' will be updated."
[2] "106-68-3 is a new odor. Data frames 'odor' and
'data.format' will be updated."
Only 'CAS' column of data has been updated.
New receptor or ORN has been added in 'ORs', please input
the expression manually.
```

preliminary version

```
[1] "DP1m is a new receptor or ORN. A new response data is
builded."
```

Not only response data (Or10a, ab5B etc.), but also the supported data "ORs", „response.range" and „weight.globNorm" were updated. The message showed that there was a new integrated response data for glumerulus "DP1m".

```
> response.range
```

	study	min	max	n_odors
1	Hallem.2006.EN	-24.00000000	294.000000	111
…				
…				
26	Galizia.2009.nmr	0.01244843	1.242804	105
27	Turner.2009.SC	-40.00000000	96.000000	47
28	Root.2007.ER	0.00000000	164.000000	5

5.2. Update response matrix

There are eight new datasets for receptors (also sensillum and glomerulus) *Or10a*, *ab5B*, *Or22a*, *Or43b*, *Or13a*, DP1m, *Or42b* and *Or59b* from "Root.2007.ER". After importing data, we can update the response matrix by merging the responses measured form different studies. We take *Or42b* as example:

```
> names(Or42b)
```

```
[1] "Class"   "Name"   "CID"   "CAS"   "Bruyne.2001.WT"   [6]
"Bruyne.2001.RR"   "Dobritsa.2003.EN"   "Kreher.2008.EN"
"Root.2007.ER"
```

After the new dataset "Root.2007.ER" has been introduced into the DoOR database, we would like to merge "Root.2007.ER" with other study as a consensus data. First, we need to check how many overlapping datapoints they share, because the data can not be merged, if the amount of overlap is less than five.

```
> apply(Or42b[,c(5:8)], 2, function(x)
{as.data.frame(na.omit(cbind(x, Or42b[,"Root.2007.ER"]))) }
)
$Bruyne.2001.WT
[1] x  V2
<0 rows> (or 0-length row.names)
```

preliminary Version

```
$Bruyne.2001.RR
      x  V2
172 147 104

$Dobritsa.2003.EN
[1] x  V2
<0 rows> (or 0-length row.names)

$Kreher.2008.EN
   x  V2
172 6 104
```

Only two datasets “Bruyne.2001.RR” and “Kreher.2008.EN” share one overlapping odor with “Root.2007.ER”, which to say, “Root.2007.ER” can not be merged into database.

To show how to merge a new measured dataset into the database, we take *Or92a* as an example.

```
> names(Or92a)

[1] "Class"            "Name"             "CID"              "CAS"
[5] "Bruyne.2001.WT"   "Bruyne.2001.RR"   "Dobritsa.2003.EN" " Galizia.2009.nmr"
```

Dataset “Galizia.2009.nmr” was measured in our lab using calcium imaging. It shares 5, 23 and 5 overlapping odors with “Bruyne.2001.WT”, “Bruyne.2001.RR” and “Dobritsa.2003.EN”, respectively.

Beside the first four information columns, there are four data columns containing datasets for *Or92a*. The entry *Or92a* in the response matrix can be updated using `updateDatabase()`. If the argument “permutation” is FALSE, the data will be merged in routine sequence; if TRUE, the sequence is chosen from testing all possible permutations. The mean correlations between all possible merged datasets (resulting from all possible merging sequences) and each original recording will be computed, the sequence with the highest correlation is the one that will be used for the actual merging.

```
> require(gregmisc) # package “gregmisc” is required for
permutation

> loadRD()

# Noted that if the permutation is equal TRUE, the update
process may take several minutes.

> updateDatabase(receptor="Or92a", permutation = TRUE)

[1] "The optimized sequence with the lowest mean MD 0.0121
is:"
```

preliminary version

```
[1] "Bruyne.2001.RR"   "Bruyne.2001.WT"
"Dobritsa.2003.EN" "Galizia.2009.nmr"
There were 50 or more warnings (use warnings() to see the
first 50)
```

```
> warnings()

Warning messages:
1: In optimize(ff2, interval.X, tol = 1e-04) :
  NA/Inf replaced by maximum positive value
2: In optimize(ff2, interval.X, tol = 1e-04) :
  NA/Inf replaced by maximum positive value
3: In optimize(ff2, interval.X, tol = 1e-04) :
…
```

The result also shows, that there were 50 or more warnings. These are due to that not all sequence combination can be merged.

5.3. Build packages

For Linux

Users might want to build their own package, if some data or functions have been introduced into DoOR. There is a manual for writing R extension available at http://cran.r-project.org/doc/manuals/. Please refer to this for a detailed explanation.

In a Linux environment, user should create a main directory for the package.

```
$ mkdir DoOR.function
$ cd DoOR.function
```

Create two directories called “R” and “man” under main directory. If users have data, an extra directory called “data” should be created as well.

```
~/$ cd DoOR.function

~/DoOR.function$ mkdir R
~/DoOR.function$ mkdir man
```

Edit a file called DESCRIPTION, you can write this file by following the extension

preliminary version

manual at http://cran.r-project.org/doc/manuals/R-exts.pdf

Then put the function files such as "default.val.R", "projectPoints.R", etc. into the directory "R". The "man" directory contains the ".Rd" files that share the same names with function such as "default.val.Rd", "projectPoints.Rd", etc. If you want to know how to write an .Rd file in detail, please see the extension manual or follow the instruction template created by `package.sekeleton`, which will be described in the Windows section.

To check whether the function and the help files .Rd have been correctly written, go back to home directory, type:

```
~/$ R CMD check DoOR.function
```

`R CMD check` can also detect the locations of mistakes. If everything is fine, you can build a package by typing:

```
~/$ R CMD build DoOR.function
```

then, a package called `DoOR.function_0.1-1.tar.gz` will be created.

For Windows

Because R was designed in a Unix environment there are some components such as compilers and programs that are missing in Windows, so that you need to download and install those components. We build the package for Windows by following the instruction at http://www.maths.bris.ac.uk/~maman/computerstuff/Rhelp/Rpackages.html#Win-Win. User can find the link to download those components including Perl, cygwin, mingwin and hhc.exe on the website.

After all components have been installed, you need to change the PATH environment variable to locate the command prompts. The "environment variable" can be found by right clicking on the "My Computer" then clicking on the "advanced" tab. Find the path and then **add**:

```
C:\Perl\bin\;C:\cygwin;C:\mingwin\bin
```

NOTE: PLEASE do not delete other path variables.

Start R, source the function and read the data by typing:

```
source("default.val.R")
source("projectPoints.R")
…

> Or22a <- read.table("Or22a.txt")
```

preliminary version

```
> Or13a <- read.table("Or13a.txt")
…
```

Specify the function names and data names respectively:

```
> Ors <- c( "Or22a","Or13a")
> funs <- c( "default.val","projectPoints")
```

build a package template:

```
> package.skeleton(list=c(Ors,funs), name="DoOR.test")

Creating directories ...
Creating DESCRIPTION ...
Creating Read-and-delete-me ...
Saving functions and data ...
Making help files ...
Done.
Further steps are described in './DoOR.test/Read-and-delete-me'.
```

Then, you will find a directory containing “man”, “data”, “R” and two files “Read-and-delete-me” and “DESCRIPTION”. Edit the “DESCRIPTION” files and all .Rd files in the “man” directory simply by filling the missing text and answering the instruction questions.

After you finished editing, you can create a package by the following:

```
C:\ Rcmd check DoOR.test

C:\ Rcmd build –-force –-binary DoOR.test
```

preliminary version

6. Acknowledgment

7. References

Meister, M. & Bonhoeffer, T., Tuning and topography in an odor map on the rat olfactory bulb. *J Neurosci,* **2001***, 21*, 1351-1360

Sachse, S.; Rappert, A. & Galizia, C. G. The spatial representation of chemical structures in the antennal lobe of honeybees: steps towards the olfactory code. **1999***, 11*, 3970-3982

Kim, H.; Golub, G. H. & Park, H. Missing value estimation for DNA microarray gene expression data: local least squares imputation. *Bioinformatics,* **2005***, 21*, 187-198

Hallem, E. A. & Carlson, J. R. Coding of odors by a receptor repertoire. *Cell,* **2006***, 125*, 143-160

Kreher, S. A.; Mathew, D.; Kim, J. & Carlson, J. R. Translation of sensory input into behavioral output via an olfactory system. *Neuron,* **2008***, 59*, 110-124

Couto, A.; Alenius, M. & Dickson, B. J. Molecular, anatomical, and functional organization of the Drosophila olfactory system. *Curr Biol,* **2005***, 15*, 1535-1547

de Bruyne, M.; Foster, K. & Carlson, J. R. Odor coding in the Drosophila antenna. *Neuron,* **2001***, 30*, 537-552

Root, C. M.; Semmelhack, J. L.; Wong, A. M.; Flores, J. & Wang, J. W. Propagation of olfactory information in Drosophila. *Proc Natl Acad Sci U S A,* **2007***, 104*, 11826-11831

http://www.maths.bris.ac.uk/~maman/computerstuff/Rhelp/Rpackages.html

R Development Core Team (2009) R: A language and environment for statistical computing. In. Vienna, Austria: R Foundation for Statistical Computing.

Schmuker M, Schneider G (2007) Processing and classification of chemical data inspired by insect olfaction. Proc Natl Acad Sci U S A 104:20285--20289.

preliminary version